# broken access control

### **Authentication vs. Authorization**

### **Authentication**

- Determines whether users are who they claim to be
- Challenges the user to validate credentials (for example, through passwords, answers to security questions, or facial recognition)
- + Usually done before authorization
- + It usually needs the user's login details
- Generally, transmits info through an ID Token

#### **Authorization**

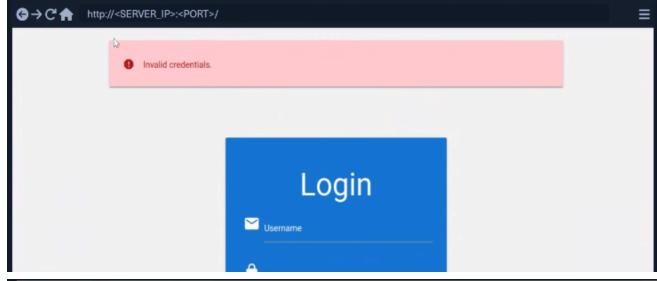
- Determines what users can and cannot access
- Verifies whether access is allowed through policies and rules
- Usually done after successful authentication
- While it needs user's privilege or security levels
- Generally, transmits info through an Access Token
- هنا اول حاجه تعرف الفرق بين authentication authorization

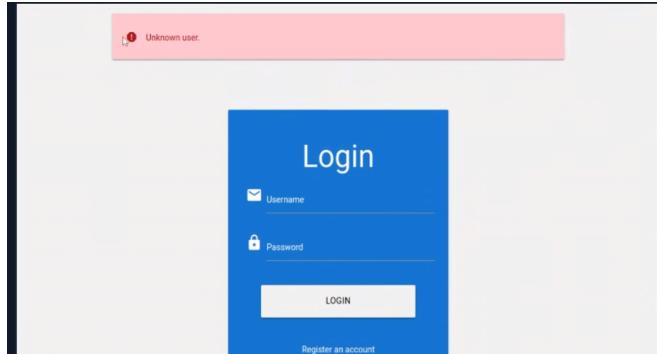
## Authentication

- بتتأكد إن المستخدم هو فعلاً الشخص الطلوب
- بتطلب من المستخدم يثبت هويته (زي الباسورد، سؤال أمان، بصمة، أو بصمة الوجه)
  - دايمًا بتحصل قبل التفويض (Authorization)
  - بتحتاج بيانات دخول المستخدم (زي اليوزر نيم والباسورد)
    - بتنقل البيانات غالبًا باستخدام ID Token

# **Authorization**

- بتحدد إيه اللي المستخدم يقدر يعمله أو يوصله
- بتراجع السياسات والقواعد علشان تشوف المستخدم له صلاحية و لا لأ.
- بتحصل بعد ما المستخدم يتأكد هويته (يعني بعد الـ Authentication).
  - بتعتمد على صلاحيات المستخدم أو مستوى الأمان الخاص بيه.
    - بتنقل البيانات غالبًا باستخدام Access Token.

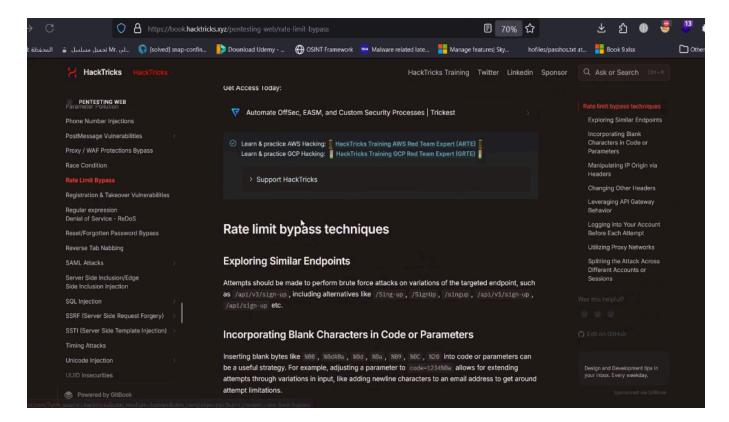




- هنا ممكن تلاحظ فرق في error
- دي ممكن تكون ثغره عشان نت كدا قدرت تعمل enumeration user

root@team:~/wordlist/SecLists/Usernames# ffuf -u http://83.136.255.196:49534 -w xato-net-10-million-usernames.txt -d "username=FUZZ&password=oifjaojad" -H "Content-Type: application/x-www-form-urlencoded" -X POST

• تستعمل fuzz عشان تعمل fuzz بالطريقه دى



- دا موقع اسمه hack tricks
- بيساعدك انه يديك حاجات تعدي بيها rate limit

Password CAPTCHA Log in ☐ Inspector ☐ Console ☐ Debugger ↑ Network {} Style Editor ☐ Performance ☐ Memory Storage 🛊 Accessibility Q. Search HTML <!DOCTYPE html> <html lang="en"> event scroll ♦ <head> ••• </head> ▼ <body> ▼ <div class="login-form"> ♥ <form action="" method="post"> <h2 class="text-center">Log in</h2> ▶ <div class="form-group"> • </div> ▶ <div class="form-group"> ··· </div> ▼ <div class="form-group"> <input class="form-control" name="CAPTCHA" type="text" placeholder="CAPTCHA" required="required"> </d1v>

• ممكن تلاقى ثغره في captha عن طريق انه هي بتكون id نفسه تعتبر ثغره

The session token is 32 characters long; thus, it seems infeasible to enumerate other users' valid sessions. However, let us send the login request multiple times and take note of the session tokens assigned by the web application. This results in the following session tokens:

2c0c58b27c71a2ec5bf2b4b6e892b9f9 2c0c58b27c71a2ec5bf2b4546092b9f9 2c0c58b27c71a2ec5bf2b497f592b9f9 2c0c58b27c71a2ec5bf2b48bcf92b9f9 2c0c58b27c71a2ec5bf2b4735e92b9f9

As we can see, all session tokens are very similar. In fact, of the 32 characters, 28 are the same for all five captured sessions.

The session tokens consist of the static string 2c0c58b27c71a2ec5bf2b4 followed by four random characters and the string 92b9f9. This reduces the effective randomness of the session tokens. Since 28 out of 32 characters are static, there are



ممكن لو لقیت session مشتابهه كدا الول اختلاف بسیط
ممكن تفكه عادی تعمله decode و تكسره